PATENT
450110-04914

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

TITLE: FACE DETECTION

INVENTORS: Robert Mark Stefan PORTER, Ratna RAMBARUTH

William S. Frommer
Registration No. 25,506
FROMMER LAWRENCE & HAUG LLP
745 Fifth Avenue
New York, New York 10151
Tel. (212) 588-0800

BACKGROUND OF THE INVENTION

Field of the Invention

This invention relates to face detection.

Description of the Prior Art

Many human-face detection algorithms have been proposed in the literature, including the use of so-called eigenfaces, face template matching, deformable template matching or neural network classification. None of these is perfect, and each generally has associated advantages and disadvantages. None gives an absolutely reliable indication that an image contains a face; on the contrary, they are all based upon a probabilistic assessment, based on a mathematical analysis of the image, of whether the image has at least a certain likelihood of containing a face. Depending on their application, the algorithms generally have the threshold likelihood value set quite high, to try to avoid false detections of faces.

In any sort of block-based analysis of a possible face, or an analysis involving a comparison between the possible face and some pre-derived data indicative of the presence of a face, there is a possibility that the algorithm will be confused by an image region which, while possibly looking nothing like a face, may possess certain image attributes to pass the comparison test. Such a region may then be assigned a high probability of containing a face, and can lead to a false-positive face detection.

It is a constant aim in this technical field to improve the reliability of face detection, including reducing the occurrence of false-positive detections.

SUMMARY OF THE INVENTION

This invention provides video face detection apparatus in which a test image from a video sequence is compared with an image property model derived from image properties of a region detected to contain a face in a preceding image in the video sequence; the apparatus comprising:

means for selecting a predetermined proportion of pixels in the region detected to contain a face in the preceding image which most closely match the image property model derived in respect of that region, thereby deriving a pixel mask; and

means for comparing pixels in the test image defined by the pixel mask with the image property model, the mask being applied at more than one image position within the test image; a face being detected in the test image at a mask position corresponding to a lowest average difference between the image property model and pixels defined by the mask at that position.

The invention provides for the use of the most appropriate portion of pixels, being that portion which most closely matches the image property model, in a face detection process. This can give a more reliable result.

It will be appreciated that the term "preceding image" and the like refer to an order of testing of the images, not necessarily to a forward temporal order of the video sequence.

Various respective aspects and features of the invention are defined in the appended claims.

BRIEF DESCRIPTION OF THE DRAWINGS

The above and other objects, features and advantages of the invention will be apparent from the following detailed description of illustrative embodiments which is to be read in connection with the accompanying drawings, in which:

Figure 1 is a schematic diagram of a general purpose computer system for use as a face detection system and/or a non-linear editing system;

Figure 2 is a schematic diagram of a video camera-recorder (camcorder) using face detection;

Figure 3 is a schematic diagram illustrating a training process;

Figure 4 is a schematic diagram illustrating a detection process;

Figure 5 schematically illustrates a feature histogram;

Figure 6 schematically illustrates a sampling process to generate eigenblocks;

Figures 7 and 8 schematically illustrates sets of eigenblocks;

Figure 9 schematically illustrates a process to build a histogram representing a block position;

Figure 10 schematically illustrates the generation of a histogram bin number;

Figure 11 schematically illustrates the calculation of a face probability;

Figures 12a to 12f are schematic examples of histograms generated using the above methods;

Figures 13a to 13g schematically illustrate so-called multiscale face detection;

Figure 14 schematically illustrates a face tracking algorithm;

Figures 15a and 15b schematically illustrate the derivation of a search area used for skin colour detection;

Figure 16 schematically illustrates a mask applied to skin colour detection;

Figures 17a to 17c schematically illustrate the use of the mask of Figure 16;

Figure 18 is a schematic distance map;

Figures 19a to 19c schematically illustrate the use of face tracking when applied to a video scene;

Figure 20 schematically illustrates a display screen of a non-linear editing system;

Figures 21a and 21b schematically illustrate clip icons; and

Figures 22a to 22c schematically illustrate a gradient pre-processing technique.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Figure 1 is a schematic diagram of a general purpose computer system for use as a face detection system and/or a non-linear editing system. The computer system comprises a processing until 10 having (amongst other conventional components) a central processing unit (CPU) 20, memory such as a random access memory (RAM) 30 and non-volatile storage such as a disc drive 40. The computer system may be connected to a network 50 such as a local area network or the Internet (or both). A keyboard 60, mouse or other user input device 70 and display screen 80 are also provided. The skilled man will appreciate that a general purpose computer system may include many other conventional parts which need not be described here.

Figure 2 is a schematic diagram of a video camera-recorder (camcorder) using face detection. The camcorder 100 comprises a lens 110 which focuses an image onto a charge coupled device (CCD) image capture device 120. The resulting image in electronic form is processed by image processing logic 130 for recording on a recording medium such as a tape cassette 140. The images captured by the device 120 are also displayed on a user display 150 which may be viewed through an eyepiece 160.

To capture sounds associated with the images, one or more microphones are used. These may be external microphones, in the sense that they are connected to the camcorder by a flexible cable, or maybe mounted on the camcorder body itself. Analogue audio signals from the microphone (s) are processed by an audio processing arrangement 170 to produce appropriate audio signals for recording on the storage medium 140.

It is noted that the video and audio signals may be recorded on the storage medium 140 in either digital form or analogue form, or even in both forms. Thus, the image processing arrangement 130 and the audio processing arrangement 170 may include a stage of analogue to digital conversion.

The camcorder user is able to control aspects of the lens 110's performance by user controls 180 which influence a lens control arrangement 190 to send electrical control

signals 200 to the lens 110. Typically, attributes such as focus and zoom are controlled in this way, but the lens aperture or other attributes may also be controlled by the user.

Two further user controls are schematically illustrated. A push button 210 is provided to initiate and stop recording onto the recording medium 140. For example, one push of the control 210 may start recording and another push may stop recording, or the control may need to be held in a pushed state for recording to take place, or one push may start recording for a certain timed period, for example five seconds. In any of these arrangements, it is technologically very straightforward to establish from the camcorder's record operation where the beginning and end of each "shot" (continuous period of recording) occurs.

The other user control shown schematically in Figure 2 is a "good shot marker" (GSM) 220, which may be operated by the user to cause "metadata" (associated data) to be stored in connection with the video and audio material on the recording medium 140, indicating that this particular shot was subjectively considered by the operator to be "good" in some respect (for example, the actors performed particularly well; the news reporter pronounced each word correctly; and so on).

The metadata may be recorded in some spare capacity (e.g. "user data") on the recording medium 140, depending on the particular format and standard in use. Alternatively, the metadata can be stored on a separate storage medium such as a removable MemoryStick^{RTM} memory (not shown), or the metadata could be stored on an external database (not shown), for example being communicated to such a database by a wireless link (not shown). The metadata can include not only the GSM information but also shot boundaries, lens attributes, alphanumeric information input by a user (e.g. on a keyboard – not shown), geographical position information from a global positioning system receiver (not shown) and so on.

So far, the description has covered a metadata-enabled camcorder. Now, the way in which face detection may be applied to such a camcorder will be described.

The camcorder includes a face detector arrangement 230. Appropriate arrangements will be described in much greater detail below, but for this part of the description it is sufficient to say that the face detector arrangement 230 receives images from the image processing arrangement 130 and detects, or attempts to detect, whether such images contain one or more faces. The face detector may output face detection data which could be in the form of a "yes/no" flag or maybe more detailed in that the data could include the image co-ordinates of the faces, such as the co-ordinates of eye positions within each detected face.

This information may be treated as another type of metadata and stored in any of the other formats described above.

As described below, face detection may be assisted by using other types of metadata within the detection process. For example, the face detector 230 receives a control signal from the lens control arrangement 190 to indicate the current focus and zoom settings of the lens 110. These can assist the face detector by giving an initial indication of the expected image size of any faces that may be present in the foreground of the image. In this regard, it is noted that the focus and zoom settings between them define the expected separation between the camcorder 100 and a person being filmed, and also the magnification of the lens 110. From these two attributes, based upon an average face size, it is possible to calculate the expected size (in pixels) of a face in the resulting image data.

A conventional (known) speech detector 240 receives audio information from the audio processing arrangement 170 and detects the presence of speech in such audio information. The presence of speech may be an indicator that the likelihood of a face being present in the corresponding images is higher than if no speech is detected.

Finally, the GSM information 220 and shot information (from the control 210) are supplied to the face detector 230, to indicate shot boundaries and those shots considered to be most useful by the user.

Of course, if the camcorder is based upon the analogue recording technique, further analogue to digital converters (ADCs) may be required to handle the image and audio information.

The present embodiment uses a face detection technique arranged as two phases. Figure 3 is a schematic diagram illustrating a training phase, and Figure 4 is a schematic diagram illustrating a detection phase.

Unlike some previously proposed face detection methods (see References 4 and 5 below), the present method is based on modelling the face in parts instead of as a whole. The parts can either be blocks centred over the assumed positions of the facial features (so-called "selective sampling") or blocks sampled at regular intervals over the face (so-called "regular sampling"). The present description will cover primarily regular sampling, as this was found in empirical tests to give the better results.

In the training phase, an analysis process is applied to a set of images known to contain faces, and (optionally) another set of images ("nonface images") known not to contain faces. The analysis process builds a mathematical model of facial and nonfacial features, against which a test image can later be compared (in the detection phase).

So, to build the mathematical model (the training process 310 of Figure 3), the basic steps are as follows:

1. From a set 300 of face images normalised to have the same eye positions, each face is sampled regularly into small blocks.

2. Attributes are calculated for each block; these attributes are explained further below.

3. The attributes are quantised to a manageable number of different values.

4. The quantised attributes are then combined to generate a single quantised value in respect of that block position.

5. The single quantised value is then recorded as an entry in a histogram, such as the schematic histogram of Figure 5. The collective histogram information 320 in respect of all of the block positions in all of the training images forms the foundation of the mathematical model of the facial features.

One such histogram is prepared for each possible block position, by repeating the above steps in respect of a large number of test face images. The test data are described further in Appendix A below. So, in a system which uses an array of 8 x 8 blocks, 64 histograms are prepared. In a later part of the processing, a test quantised attribute is compared with the histogram data; the fact that a whole histogram is used to model the data means that no assumptions have to be made about whether it follows a parameterised distribution, e.g. Gaussian or otherwise. To save data storage space (if needed), histograms which are similar can be merged so that the same histogram can be reused for different block positions.

In the detection phase, to apply the face detector to a test image 350, successive windows in the test image are processed 340 as follows:

6. The window is sampled regularly as a series of blocks, and attributes in respect of each block are calculated and quantised as in stages 1-4 above.

7. Corresponding "probabilities" for the quantised attribute values for each block position are looked up from the corresponding histograms. That is to say, for each block position, a respective quantised attribute is generated and is compared with a histogram previously generated in respect of that block position. The way in which the histograms give rise to "probability" data will be described below.

8. All the probabilities obtained above are multiplied together to form a final probability which is compared against a threshold in order to classify the window as "face" or "nonface". It will be appreciated that the detection result of "face" or "nonface" is a probability-based measure rather than an absolute detection. Sometimes, an image not

containing a face may be wrongly detected as "face", a so-called false positive. At other times, an image containing a face may be wrongly detected as "nonface", a so-called false negative. It is an aim of any face detection system to reduce the proportion of false positives and the proportion of false negatives, but it is of course understood that to reduce these proportions to zero is difficult, if not impossible, with current technology.

As mentioned above, in the training phase, a set of "nonface" images can be used to generate a corresponding set of "nonface" histograms. Then, to achieve detection of a face, the "probability" produced from the nonface histograms may be compared with a separate threshold, so that the probability has to be under the threshold for the test window to contain a face. Alternatively, the ratio of the face probability to the nonface probability could be compared with a threshold.

Extra training data may be generated by applying "synthetic variations" 330 to the original training set, such as variations in position, orientation, size, aspect ratio, background scenery, lighting intensity and frequency content.

The derivation of attributes and their quantisation will now be described. In the present technique, attributes are measured with respect to so-called eigenblocks, which are core blocks (or eigenvectors) representing different types of block which may be present in the windowed image. The generation of eigenblocks will first be described with reference to Figure 6.

Eigenblock creation

The attributes in the present embodiment are based on so-called eigenblocks. The eigenblocks were designed to have good representational ability of the blocks in the training set. Therefore, they were created by performing principal component analysis on a large set of blocks from the training set. This process is shown schematically in Figure 6 and described in more detail in Appendix B.

Training the System

Experiments were performed with two different sets of training blocks.

Eigenblock set I

Initially, a set of blocks were used that were taken from 25 face images in the training set. The 16x16 blocks were sampled every 16 pixels and so were non-overlapping. This

sampling is shown in Figure 6. As can be seen, 16 blocks are generated from each 64x64 training image. This leads to a total of 400 training blocks overall.

The first 10 eigenblocks generated from these training blocks are shown in Figure 7.

Eigenblock set II

A second set of eigenblocks was generated from a much larger set of training blocks. These blocks were taken from 500 face images in the training set. In this case, the 16x16 blocks were sampled every 8 pixels and so overlapped by 8 pixels. This generated 49 blocks from each 64x64 training image and led to a total of 24,500 training blocks.

The first 12 eigenblocks generated from these training blocks are shown in Figure 8.

Empirical results show that eigenblock set II gives slightly better results than set I. This is because it is calculated from a larger set of training blocks taken from face images, and so is perceived to be better at representing the variations in faces. However, the improvement in performance is not large.

Building the Histograms

A histogram was built for each sampled block position within the 64x64 face image. The number of histograms depends on the block spacing. For example, for block spacing of 16 pixels, there are 16 possible block positions and thus 16 histograms are used.

The process used to build a histogram representing a single block position is shown in Figure 9. The histograms are created using a large training set 400 of M face images. For each face image, the process comprises:

- Extracting 410 the relevant block from a position (i,j) in the face image.
- Calculating the eigenblock-based attributes for the block, and determining the relevant bin number 420 from these attributes.
- Incrementing the relevant bin number in the histogram 430.

This process is repeated for each of M images in the training set, to create a histogram that gives a good representation of the distribution of frequency of occurrence of the attributes. Ideally, M is very large, e.g. several thousand. This can more easily be achieved by using a training set made up of a set of original faces and several hundred synthetic variations of each original face.

Generating the histogram bin number

A histogram bin number is generated from a given block using the following process, as shown in Figure 10. The 16x16 block 440 is extracted from the 64x64 window or face image. The block is projected onto the set 450 of A eigenblocks to generate a set of "eigenblock weights". These eigenblock weights are the "attributes" used in this implementation. They have a range of -1 to $+1$. **This process is described in more detail in Appendix B.**Each weight is quantised into a fixed number of levels, L, to produce a set of quantised attributes 470, $w_i, i = 1..A$. The quantised weights are combined into a single value as follows:

$$h = w_1 L^{A-1} + w_2 L^{A-2} + w_3 L^{A-3} + ... + w_{A-1} L^1 + w_A L^0$$

where the value generated, h, is the histogram bin number 480. Note that the total number of bins in the histogram is given by L^A.

The bin "contents", i.e. the frequency of occurrence of the set of attributes giving rise to that bin number, may be considered to be a probability value if it is divided by the number of training images M. However, because the probabilities are compared with a threshold, there is in fact no need to divide through by M as this value would cancel out in the calculations. So, in the following discussions, the bin "contents" will be referred to as "probability values", and treated as though they are probability values, even though in a strict sense they are in fact frequencies of occurrence.

The above process is used both in the training phase and in the detection phase.

Face Detection Phase

The face detection process involves sampling the test image with a moving 64x64 window and calculating a face probability at each window position.

The calculation of the face probability is shown in Figure 11. For each block position in the window, the block's bin number 490 is calculated as described in the previous section. Using the appropriate histogram 500 for the position of the block, each bin number is looked up and the probability 510 of that bin number is determined. The sum 520 of the logs of these probabilities is then calculated across all the blocks to generate a face probability value, P_{face} (otherwise referred to as a log likelihood value).

This process generates a probability "map" for the entire test image. In other words, a probability value is derived in respect of each possible window centre position across the

image. The combination of all of these probability values into a rectangular (or whatever) shaped array is then considered to be a probability "map" corresponding to that image.

This map is then inverted, so that the process of finding a face involves finding minima in the inverted map. A so-called distance-based technique is used. This technique can be summarised as follows: The map (pixel) position with the smallest value in the inverted probability map is chosen. If this value is larger than a threshold (TD), no more faces are chosen. This is the termination criterion. Otherwise a face-sized block corresponding to the chosen centre pixel position is blanked out (i.e. omitted from the following calculations) and the candidate face position finding procedure is repeated on the rest of the image until the termination criterion is reached.

Nonface method

The nonface model comprises an additional set of histograms which represent the probability distribution of attributes in nonface images. The histograms are created in exactly the same way as for the face model, except that the training images contain examples of nonfaces instead of faces.

During detection, two log probability values are computed, one using the face model and one using the nonface model. These are then combined by simply subtracting the nonface probability from the face probability:

$$P_{combined} = P_{face} - P_{nonface}$$

$P_{combined}$ is then used instead of P_{face} to produce the probability map (before inversion).

Note that the reason that $P_{nonface}$ is subtracted from P_{face} is because these are log probability values.

Histogram Examples

Figures 12a to 12f show some examples of histograms generated by the training process described above.

Figures 12a, 12b and 12c are derived from a training set of face images, and Figures 12d, 12e and 12f are derived from a training set of nonface images. In particular:

	Face histograms	Nonface histograms
Whole histogram	Figure 12a	Figure 12d
Zoomed onto the main peaks at about h=1500	Figure 12b	Figure 12e
A further zoom onto the region about h=1570	Figure 12c	Figure 12f

It can clearly be seen that the peaks are in different places in the face histogram and the nonface histograms.

Multiscale face detection

In order to detect faces of different sizes in the test image, the test image is scaled by a range of factors and a distance (i.e. probability) map is produced for each scale. In Figures 13a to 13c the images and their corresponding distance maps are shown at three different scales. The method gives the best response (highest probability, or minimum distance) for the large (central) subject at the smallest scale (Fig 13a) and better responses for the smaller subject (to the left of the main figure) at the larger scales. (A darker colour on the map represents a lower value in the inverted map, or in other words a higher probability of there being a face).Candidate face positions are extracted across different scales by first finding the position which gives the best response over all scales. That is to say, the highest probability (lowest distance) is established amongst all of the probability maps at all of the scales. This candidate position is the first to be labelled as a face. The window centred over that face position is then blanked out from the probability map at each scale. The size of the window blanked out is proportional to the scale of the probability map.

Examples of this scaled blanking-out process are shown in Figures 13a to 13c. In particular, the highest probability across all the maps is found at the left hand side of the largest scale map (Figure 13c). An area 530 corresponding to the presumed size of a face is blanked off in Figure 13c. Corresponding, but scaled, areas 532, 534 are blanked off in the smaller maps.

Areas larger than the test window may be blanked off in the maps, to avoid overlapping detections. In particular, an area equal to the size of the test window surrounded

by a border half as wide/long as the test window is appropriate to avoid such overlapping detections.

Additional faces are detected by searching for the next best response and blanking out the corresponding windows successively.

The intervals allowed between the scales processed are influenced by the sensitivity of the method to variations in size. It was found in this preliminary study of scale invariance that the method is not excessively sensitive to variations in size as faces which gave a good response at a certain scale often gave a good response at adjacent scales as well.

The above description refers to detecting a face even though the size of the face in the image is not known at the start of the detection process. Another aspect of multiple scale face detection is the use of two or more parallel detections at different scales to validate the detection process. This can have advantages if, for example, the face to be detected is partially obscured, or the person is wearing a hat etc.

Figures 13d to 13g schematically illustrate this process. During the training phase, the system is trained on windows (divided into respective blocks as described above) which surround the whole of the test face (Figure 13d) to generate "full face" histogram data and also on windows at an expanded scale so that only a central area of the test face is included (Figure 13e) to generate "zoomed in" histogram data. This generates two sets of histogram data. One set relates to the "full face" windows of Figure 13d, and the other relates to the "central face area" windows of Figure 13e.

During the detection phase, for any given test window 536, the window is applied to two different scalings of the test image so that in one (Figure 13f) the test window surrounds the whole of the expected size of a face, and in the other (Figure 13g) the test window encompasses the central area of a face at that expected size. These are each processed as described above, being compared with the respective sets of histogram data appropriate to the type of window. The log probabilities from each parallel process are added before the comparison with a threshold is applied.

Putting both of these aspects of multiple scale face detection together leads to a particularly elegant saving in the amount of data that needs to be stored.

In particular, in these embodiments the multiple scales for the arrangements of Figures 13a to 13c are arranged in a geometric sequence. In the present example, each scale in the sequence is a factor of $\sqrt[4]{2}$ different to the adjacent scale in the sequence. Then, for the parallel detection described with reference to Figures 13d to 13g, the larger scale, central

area, detection is carried out at a scale 3 steps higher in the sequence, that is, $2^{3/4}$ times larger than the "full face" scale, using attribute data relating to the scale 3 steps higher in the sequence. So, apart from at extremes of the range of multiple scales, the geometric progression means that the parallel detection of Figures 13d to 13g can always be carried out using attribute data generated in respect of another multiple scale three steps higher in the sequence.

The two processes (multiple scale detection and parallel scale detection) can be combined in various ways. For example, the multiple scale detection process of Figures 13a to 13c can be applied first, and then the parallel scale detection process of Figures 13d to 13g can be applied at areas (and scales) identified during the multiple scale detection process. However, a convenient and efficient use of the attribute data may be achieved by:

- deriving attributes in respect of the test window at each scale (as in Figures 13a to 13c)
- comparing those attributes with the "full face" histogram data to generate a "full face" set of distance maps
- comparing the attributes with the "zoomed in" histogram data to generate a "zoomed in" set of distance maps
- for each scale n, combining the "full face" distance map for scale n with the "zoomed in" distance map for scale n+3
- deriving face positions from the combined distance maps as described above with reference to Figures 13a to 13c

Further parallel testing can be performed to detect different poses, such as looking straight ahead, looking partly up, down, left, right etc. Here a respective set of histogram data is required and the results are preferably combined using a "max" function, that is, the pose giving the highest probability is carried forward to thresholding, the others being discarded.

Face Tracking

A face tracking algorithm will now be described. The tracking algorithm aims to improve face detection performance in image sequences.

The initial aim of the tracking algorithm is to detect every face in every frame of an image sequence. However, it is recognised that sometimes a face in the sequence may not be detected. In these circumstances, the tracking algorithm may assist in interpolating across the missing face detections.

Ultimately, the goal of face tracking is to be able to output some useful metadata from each set of frames belonging to the same scene in an image sequence. This might include:

- Number of faces.
- "Mugshot" (a colloquial word for an image of a person's face, derived from a term referring to a police file photograph) of each face.
- Frame number at which each face first appears.
- Frame number at which each face last appears.
- Identity of each face (either matched to faces seen in previous scenes, or matched to a face database) – this requires some face recognition also.

The tracking algorithm uses the results of the face detection algorithm, run independently on each frame of the image sequence, as its starting point. Because the face detection algorithm may sometimes miss (not detect) faces, some method of interpolating the missing faces is useful. To this end, a Kalman filter was used to predict the next position of the face and a skin colour matching algorithm was used to aid tracking of faces. In addition, because the face detection algorithm often gives rise to false acceptances, some method of rejecting these is also useful.

The algorithm is shown schematically in Figure 14.

The algorithm will be described in detail below, but in summary, input video data 545 (representing the image sequence) is supplied to a face detector of the type described in this application, and a skin colour matching detector 550. The face detector attempts to detect one or more faces in each image. When a face is detected, a Kalman filter 560 is established to track the position of that face. The Kalman filter generates a predicted position for the same face in the next image in the sequence. An eye position comparator 570, 580 detects whether the face detector 540 detects a face at that position (or within a certain threshold distance of that position) in the next image. If this is found to be the case, then that detected face position is used to update the Kalman filter and the process continues.

If a face is not detected at or near the predicted position, then a skin colour matching method 550 is used. This is a less precise face detection technique which is set up to have a lower threshold of acceptance than the face detector 540, so that it is possible for the skin colour matching technique to detect (what it considers to be) a face even when the face detector cannot make a positive detection at that position. If a "face" is detected by skin

colour matching, its position is passed to the Kalman filter as an updated position and the process continues.

If no match is found by either the face detector 450 or the skin colour detector 550, then the predicted position is used to update the Kalman filter.

All of these results are subject to acceptance criteria (see below). So, for example, a face that is tracked throughout a sequence on the basis of one positive detection and the remainder as predictions, or the remainder as skin colour detections, will be rejected.

A separate Kalman filter is used to track each face in the tracking algorithm.

In order to use a Kalman filter to track a face, a state model representing the face must be created. In the model, the position of each face is represented by a 4-dimensional vector containing the co-ordinates of the left and right eyes, which in turn are derived by a predetermined relationship to the centre position of the window and the scale being used:

$$p(k) = \begin{bmatrix} FirstEyeX \\ FirstEyeY \\ SecondEyeX \\ SecondEyeY \end{bmatrix}$$

where k is the frame number.

The current state of the face is represented by its position, velocity and acceleration, in a 12-dimensional vector:

$$\hat{z}(k) = \begin{bmatrix} p(k) \\ \dot{p}(k) \\ \ddot{p}(k) \end{bmatrix}$$

First Face Detected

The tracking algorithm does nothing until it receives a frame with a face detection result indicating that there is a face present.

A Kalman filter is then initialised for each detected face in this frame. Its state is initialised with the position of the face, and with zero velocity and acceleration:

$$\hat{z}_a(k) = \begin{bmatrix} p(k) \\ 0 \\ 0 \end{bmatrix}$$

It is also assigned some other attributes: the state model error covariance, Q and the observation error covariance, R. The error covariance of the Kalman filter, P, is also

initialised. These parameters are described in more detail below. At the beginning of the following frame, and every subsequent frame, a Kalman filter prediction process is carried out.

Kalman Filter Prediction Process

For each existing Kalman filter, the next position of the face is predicted using the standard Kalman filter prediction equations shown below. The filter uses the previous state (at frame k-1) and some other internal and external variables to estimate the current state of the filter (at frame k).

State prediction equation: $\hat{z}_b(k) = \Phi(k,k-1)\hat{z}_a(k-1)$

Covariance prediction equation: $P_b(k) = \Phi(k,k-1)P_a(k-1)\Phi(k,k-1)^T + Q(k)$

where $\hat{z}_b(k)$ denotes the state before updating the filter for frame k, $\hat{z}_a(k-1)$ denotes the state *a*fter updating the filter for frame k-1 (or the initialised state if it is a new filter), and $\Phi(k,k-1)$ is the state transition matrix. Various state transition matrices were experimented with, as described below. Similarly, $P_b(k)$ denotes the filter's error covariance *b*efore updating the filter for frame k and $P_a(k-1)$ denotes the filter's error covariance *a*fter updating the filter for the previous frame (or the initialised value if it is a new filter). $P_b(k)$ can be thought of as an internal variable in the filter that models its accuracy.

$Q(k)$ is the error covariance of the state model. A high value of $Q(k)$ means that the predicted values of the filter's state (i.e. the face's position) will be assumed to have a high level of error. By tuning this parameter, the behaviour of the filter can be changed and potentially improved for face detection.

State Transition Matrix

The state transition matrix, $\Phi(k,k-1)$, determines how the prediction of the next state is made. Using the equations for motion, the following matrix can be derived for $\Phi(k,k-1)$:

$$\Phi(k,k-1)=\begin{bmatrix} I_4 & I_4\Delta t & \frac{1}{2}I_4(\Delta t)^2 \\ O_4 & I_4 & I_4\Delta t \\ O_4 & O_4 & I_4 \end{bmatrix}$$

where O_4 is a 4x4 zero matrix and I_4 is a 4x4 identity matrix. Δt can simply be set to 1 (i.e. units of t are frame periods).

This state transition matrix models position, velocity and acceleration. However, it was found that the use of acceleration tended to make the face predictions accelerate towards the edge of the picture when no face detections were available to correct the predicted state. Therefore, a simpler state transition matrix without using acceleration was preferred:

$$\Phi(k,k-1)=\begin{bmatrix} I_4 & I_4\Delta t & O_4 \\ O_4 & I_4 & O_4 \\ O_4 & O_4 & O_4 \end{bmatrix}$$

The predicted eye positions of each Kalman filter, $\hat{z}_b(k)$, are compared to all face detection results in the current frame (if there are any). If the distance between the eye positions is below a given threshold, then the face detection can be assumed to belong to the same face as that being modelled by the Kalman filter. The face detection result is then treated as an observation, $y(k)$, of the face's current state:

$$y(k)=\begin{bmatrix} p(k) \\ 0 \\ 0 \end{bmatrix}$$

where $p(k)$ is the position of the eyes in the face detection result. This observation is used during the Kalman filter update stage to help correct the prediction.

Skin Colour Matching

Skin colour matching is not used for faces that successfully match face detection results. Skin colour matching is only performed for faces whose position has been predicted by the Kalman filter but have no matching face detection result in the current frame, and therefore no observation data to help update the Kalman filter.

In a first technique, for each face, an elliptical area centred on the face's previous position is extracted from the previous frame. An example of such an area 600 within the face window 610 is shown schematically in Figure 16. A colour model is seeded using the

chrominance data from this area to produce an estimate of the mean and covariance of the Cr and Cb values, based on a Gaussian model.

An area around the predicted face position in the current frame is then searched and the position that best matches the colour model, again averaged over an elliptical area, is selected. If the colour match meets a given similarity criterion, then this position is used as an observation, $y(k)$, of the face's current state in the same way described for face detection results in the previous section.

Figures 15a and 15b schematically illustrate the generation of the search area. In particular, Figure 15a schematically illustrates the predicted position 620 of a face within the next image 630. In skin colour matching, a search area 640 surrounding the predicted position 620 in the next image is searched for the face.

If the colour match does not meet the similarity criterion, then no reliable observation data is available for the current frame. Instead, the predicted state, $\hat{z}_b(k)$ is used as the observation:

$$y(k) = \hat{z}_b(k)$$

The skin colour matching methods described above use a simple Gaussian skin colour model. The model is seeded on an elliptical area centred on the face in the previous frame, and used to find the best matching elliptical area in the current frame. However, to provide a potentially better performance, two further methods were developed: a colour histogram method and a colour mask method. These will now be described.

Colour Histogram Method

In this method, instead of using a Gaussian to model the distribution of colour in the tracked face, a colour histogram is used.

For each tracked face in the previous frame, a histogram of Cr and Cb values within a square window around the face is computed. To do this, for each pixel the Cr and Cb values are first combined into a single value. A histogram is then computed that measures the frequency of occurrence of these values in the whole window. Because the number of combined Cr and Cb values is large (256x256 possible combinations), the values are quantised before the histogram is calculated.

Having calculated a histogram for a tracked face in the previous frame, the histogram is used in the current frame to try to estimate the most likely new position of the face by finding the area of the image with the most similar colour distribution. As shown

schematically in Figures 15a and 15b, this is done by calculating a histogram in exactly the same way for a range of window positions within a search area of the current frame. This search area covers a given area around the predicted face position. The histograms are then compared by calculating the mean squared error (MSE) between the original histogram for the tracked face in the previous frame and each histogram in the current frame. The estimated position of the face in the current frame is given by the position of the minimum MSE.

Various modifications may be made to this algorithm, including:

- Using three channels (Y, Cr and Cb) instead of two (Cr, Cb).
- Varying the number of quantisation levels.
- Dividing the window into blocks and calculating a histogram for each block. In this way, the colour histogram method becomes positionally dependent. The MSE between each pair of histograms is summed in this method.
- Varying the number of blocks into which the window is divided.
- Varying the blocks that are actually used – e.g. omitting the outer blocks which might only partially contain face pixels.

For the test data used in empirical trials of these techniques, the best results were achieved using the following conditions, although other sets of conditions may provide equally good or better results with different test data:

- 3 channels (Y, Cr and Cb).
- 8 quantisation levels for each channel (i.e. histogram contains 8x8x8 = 512 bins).
- Dividing the windows into 16 blocks.
- Using all 16 blocks.

Colour Mask Method

This method is based on the method first described above. It uses a Gaussian skin colour model to describe the distribution of pixels in the face.

In the method first described above, an elliptical area centred on the face is used to colour match faces, as this may be perceived to reduce or minimise the quantity of background pixels which might degrade the model.

In the present colour mask model, a similar elliptical area is still used to seed a colour model on the original tracked face in the previous frame, for example by applying the mean and covariance of RGB or YCrCb to set parameters of a Gaussian model (or alternatively, a

default colour model such as a Gaussian model can be used, see below). However, it is not used when searching for the best match in the current frame. Instead, a mask area is calculated based on the distribution of pixels in the original face window from the previous frame. The mask is calculated by finding the 50% of pixels in the window which best match the colour model. An example is shown in Figures 17a to 17c. In particular, Figure 17a schematically illustrates the initial window under test; Figure 17b schematically illustrates the elliptical window used to seed the colour model; and Figure 17c schematically illustrates the mask defined by the 50% of pixels which most closely match the colour model.

To estimate the position of the face in the current frame, a search area around the predicted face position is searched (as before) and the "distance" from the colour model is calculated for each pixel. The "distance" refers to a difference from the mean, normalised in each dimension by the variance in that dimension. An example of the resultant distance image is shown in Figure 18. For each position in this distance map (or for a reduced set of sampled positions to reduce computation time), the pixels of the distance image are averaged over a mask-shaped area. The position with the lowest averaged distance is then selected as the best estimate for the position of the face in this frame.

This method thus differs from the original method in that a mask-shaped area is used in the distance image, instead of an elliptical area. This allows the colour match method to use both colour and shape information.

Two variations are proposed and were implemented in empirical trials of the techniques:

(a) Gaussian skin colour model is seeded using the mean and covariance of Cr and Cb from an elliptical area centred on the tracked face in the previous frame.

(b) A default Gaussian skin colour model is used, both to calculate the mask in the previous frame and calculate the distance image in the current frame.

The use of Gaussian skin colour models will now be described further. A Gaussian model for the skin colour class is built using the chrominance components of the YCbCr colour space. The similarity of test pixels to the skin colour class can then be measured. This method thus provides a skin colour likelihood estimate for each pixel, independently of the eigenface-based approaches.

Let w be the vector of the CbCr values of a test pixel. The probability of w belonging to the skin colour class S is modelled by a two-dimensional Gaussian:

$$p(w \mid S) = \frac{\exp\left[-\frac{1}{2}(w-\mu_s)\Sigma_s^{-1}(w-\mu_s)\right]}{2\pi|\Sigma_s|^{\frac{1}{2}}}$$

where the mean μ_s and the covariance matrix Σ_s of the distribution are (previously) estimated from a training set of skin colour values.

Skin colour detection is not considered to be an effective face detector when used on its own. This is because there can be many areas of an image that are similar to skin colour but are not necessarily faces, for example other parts of the body. However, it can be used to improve the performance of the eigenblock-based approaches by using a combined approach as described in respect of the present face tracking system. The decisions made on whether to accept the face detected eye positions or the colour matched eye positions as the observation for the Kalman filter, or whether no observation was accepted, are stored. These are used later to assess the ongoing validity of the faces modelled by each Kalman filter.

Kalman Filter Update Step

The update step is used to determine an appropriate output of the filter for the current frame, based on the state prediction and the observation data. It also updates the internal variables of the filter based on the error between the predicted state and the observed state.

The following equations are used in the update step:

Kalman gain equation $$K(k) = P_b(k)H^T(k)\left(H(k)P_b(k)H^T(k) + R(k)\right)^{-1}$$

State update equation $$\hat{z}_a(k) = \hat{z}_b(k) + K(k)\left[y(k) - H(k)\hat{z}_b(k)\right]$$

Covariance update equation $$P_a(k) = P_b(k) - K(k)H(k)P_b(k)$$

Here, $K(k)$ denotes the Kalman gain, another variable internal to the Kalman filter. It is used to determine how much the predicted state should be adjusted based on the observed state, $y(k)$.

$H(k)$ is the observation matrix. It determines which parts of the state can be observed. In our case, only the position of the face can be observed, not its velocity or acceleration, so the following matrix is used for $H(k)$:

$$H(k) = \begin{bmatrix} I_4 & O_4 & O_4 \\ O_4 & O_4 & O_4 \\ O_4 & O_4 & O_4 \end{bmatrix}$$

$R(k)$ is the error covariance of the observation data. In a similar way to $Q(k)$, a high value of $R(k)$ means that the observed values of the filter's state (i.e. the face detection results or colour matches) will be assumed to have a high level of error. By tuning this parameter, the behaviour of the filter can be changed and potentially improved for face detection. For our experiments, a large value of $R(k)$ relative to $Q(k)$ was found to be suitable (this means that the predicted face positions are treated as more reliable than the observations). Note that it is permissible to vary these parameters from frame to frame. Therefore, an interesting future area of investigation may be to adjust the relative values of $R(k)$ and $Q(k)$ depending on whether the observation is based on a face detection result (reliable) or a colour match (less reliable).

For each Kalman filter, the updated state, $\hat{z}_a(k)$, is used as the final decision on the position of the face. This data is output to file and stored.

Unmatched face detection results are treated as new faces. A new Kalman filter is initialised for each of these. Faces are removed which:

- Leave the edge of the picture and/or
- Have a lack of ongoing evidence supporting them (when there is a high proportion of observations based on Kalman filter predictions rather than face detection results or colour matches).

For these faces, the associated Kalman filter is removed and no data is output to file. As an optional difference from this approach, where a face is detected to leave the picture, the tracking results up to the frame before it leaves the picture may be stored and treated as valid face tracking results (providing that the results meet any other criteria applied to validate tracking results).

These rules may be formalised and built upon by bringing in some additional variables:

prediction_acceptance_ratio_threshold	If, during tracking a given face, the proportion of accepted Kalman predicted face positions exceeds this threshold, then the tracked face is rejected. This is currently set to 0.8.

detection_acceptance_ratio_threshold During a final pass through all the frames, if for a given face the proportion of accepted face detections falls below this threshold, then the tracked face is rejected.
This is currently set to 0.08.

min_frames During a final pass through all the frames, if for a given face the number of occurrences is less than min_frames, the face is rejected. This is only likely to occur near the end of a sequence.
min_frames is currently set to 5.

final_prediction_acceptance_ratio_threshold and *min_frames2* During a final pass through all the frames, if for a given tracked face the number of occurrences is less than min_frames2 AND the proportion of accepted Kalman predicted face positions exceeds the final_prediction_acceptance_ratio_threshold, the face is rejected. Again, this is only likely to occur near the end of a sequence.
final_prediction_acceptance_ratio_threshold is currently set to 0.5 and min_frames2 is currently set to 10.

min_eye_spacing Additionally, faces are now removed if they are tracked such that the eye spacing is decreased below a given minimum distance. This can happen if the Kalman filter falsely believes the eye distance is becoming smaller and there is no other evidence, e.g. face detection results, to correct this assumption. If uncorrected, the eye distance would eventually become zero. As an optional alternative, a minimum or lower limit eye separation can be forced, so that if the

detected eye separation reduces to the minimum eye separation, the detection process continues to search for faces having that eye separation, but not a smaller eye separation.

It is noted that the tracking process is not limited to tracking through a video sequence in a forward temporal direction. Assuming that the image data remain accessible (i.e. the process is not real-time, or the image data are buffered for temporary continued use), the entire tracking process could be carried out in a reverse temporal direction. Or, when a first face detection is made (often part-way through a video sequence) the tracking process could be initiated in both temporal directions. As a further option, the tracking process could be run in both temporal directions through a video sequence, with the results being combined so that (for example) a tracked face meeting the acceptance criteria is included as a valid result whichever direction the tracking took place.

In the tracking system shown schematically in Figure 14, three further features are included.

Shot boundary data 560 (from metadata associated with the image sequence under test; or metadata generated within the camera of Figure 2) defines the limits of each contiguous "shot" within the image sequence. The Kalman filter is reset at shot boundaries, and is not allowed to carry a prediction over to a subsequent shot, as the prediction would be meaningless.

User metadata 542 and camera setting metadata 544 are supplied as inputs to the face detector 540. These may also be used in a non-tracking system. Examples of the camera setting metadata were described above. User metadata may include information such as:

- type of programme (e.g. news, interview, drama)
- script information such as specification of a "long shot" , "medium close-up" etc (particular types of camera shot leading to an expected sub-range of face sizes), how many people involved in each shot (again leading to an expected sub-range of face sizes) and so on
- sports-related information – sports are often filmed from fixed camera positions using standard views and shots. By specifying these in the metadata, again a sub-range of face sizes can be derived

The type of programme is relevant to the type of face which may be expected in the images or image sequence. For example, in a news programme, one would expect to see a single face for much of the image sequence, occupying an area of (say) 10% of the screen. The detection of faces at different scales can be weighted in response to this data, so that faces of about this size are given an enhanced probability. Another alternative or additional approach is that the search range is reduced, so that instead of searching for faces at all possible scales, only a subset of scales is searched. This can reduce the processing requirements of the face detection process. In a software-based system, the software can run more quickly and/or on a less powerful processor. In a hardware-based system (including for example an application-specific integrated circuit (ASIC) or field programmable gate array (FPGA) system) the hardware needs may be reduced.

The other types of user metadata mentioned above may also be applied in this way. The "expected face size" sub-ranges may be stored in a look-up table held in the memory 30, for example.

As regards camera metadata, for example the current focus and zoom settings of the lens 110, these can also assist the face detector by giving an initial indication of the expected image size of any faces that may be present in the foreground of the image. In this regard, it is noted that the focus and zoom settings between them define the expected separation between the camcorder 100 and a person being filmed, and also the magnification of the lens 110. From these two attributes, based upon an average face size, it is possible to calculate the expected size (in pixels) of a face in the resulting image data, leading again to a sub-range of sizes for search or a weighting of the expected face sizes.

Advantages of the tracking algorithm

The face tracking technique has three main benefits:

- It allows missed faces to be filled in by using Kalman filtering and skin colour tracking in frames for which no face detection results are available. This increases the true acceptance rate across the image sequence.
- It provides face linking: by successfully tracking a face, the algorithm automatically knows whether a face detected in a future frame belongs to the same person or a different person. Thus, scene metadata can easily be generated from this algorithm, comprising the number of faces in the scene, the frames for which they are present and providing a representative mugshot of each face.

- False face detections tend to be rejected, as such detections tend not to carry forward between images.

Figures 19a to 19c schematically illustrate the use of face tracking when applied to a video scene.

In particular, Figure 19a schematically illustrates a video scene 800 comprising successive video images (e.g. fields or frames) 810.

In this example, the images 810 contain one or more faces. In particular all of the images 810 in the scene include a face A, shown at an upper left-hand position within the schematic representation of the image 810. Also, some of the images include a face B shown schematically at a lower right hand position within the schematic representations of the images 810.

A face tracking process is applied to the scene of Figure 19a. Face A is tracked reasonably successfully throughout the scene. In one image 820 the face is not tracked by a direct detection, but the skin colour matching techniques and the Kalman filtering techniques described above mean that the detection can be continuous either side of the "missing" image 820. The representation of Figure 19b indicates the detected probability of a face being present in each of the images. It can be seen that the probability is highest at an image 830, and so the part 840 of the image detected to contain face A is used as a "picture stamp" in respect of face A. Picture stamps will be described in more detail below.

Similarly, face B is detected with different levels of confidence, but an image 850 gives rise to the highest detected probability of face B being present. Accordingly, the part of the corresponding image detected to contain face B (part 860) is used as a picture stamp for face B within that scene. (Alternatively, of course, a wider section of the image, or even the whole image, could be used as the picture stamp).

Figure 20 schematically illustrates a display screen of a non-linear editing system.

Non-linear editing systems are well established and are generally implemented as software programs running on general purpose computing systems such as the system of Figure 1. These editing systems allow video, audio and other material to be edited to an output media product in a manner which does not depend on the order in which the individual media items (e.g. video shots) were captured.

The schematic display screen of Figure 20 includes a viewer area 900, in which video clips be may viewed, a set of clip icons 910, to be described further below and a "timeline" 920 including representations of edited video shots 930, each shot optionally containing a picture stamp 940 indicative of the content of that shot.

At one level, the face picture stamps derived as described with reference to Figures 19a to 19c could be used as the picture stamps 940 of each edited shot so, within the edited length of the shot, which may be shorter than the originally captured shot, the picture stamp representing a face detection which resulted in the highest face probability value can be inserted onto the time line to show a representative image from that shot. The probability values may be compared with a threshold, possibly higher than the basic face detection threshold, so that only face detections having a high level of confidence are used to generate picture stamps in this way. If more than one face is detected in the edited shot, the face with the highest probability may be displayed, or alternatively more than one face picture stamp may be displayed on the time line.

Time lines in non-linear editing systems are usually capable of being scaled, so that the length of line corresponding to the full width of the display screen can represent various different time periods in the output media product. So, for example, if a particular boundary between two adjacent shots is being edited to frame accuracy, the time line may be "expanded" so that the width of the display screen represents a relatively short time period in the output media product. On the other hand, for other purposes such as visualising an overview of the output media product, the time line scale may be contracted so that a longer time period may be viewed across the width of the display screen. So, depending on the level of expansion or contraction of the time line scale, there may be less or more screen area available to display each edited shot contributing to the output media product.

In an expanded time line scale, there may well be more than enough room to fit one picture stamp (derived as shown in Figures 19a to 19c) for each edited shot making up the output media product. However, as the time line scale is contracted, this may no longer be possible. In such cases, the shots may be grouped together in to "sequences", where each sequence is such that it is displayed at a display screen size large enough to accommodate a phase picture stamp. From within the sequence, then, the face picture stamp having the highest corresponding probability value is selected for display. If no face is detected within a sequence, an arbitrary image, or no image, can be displayed on the timeline.

Figure 20 also shows schematically two "face timelines" 925, 935. These scale with the "main" timeline 920. Each face timeline relates to a single tracked face, and shows the portions of the output edited sequence containing that tracked face. It is possible that the user may observe that certain faces relate to the same person but have not been associated with one another by the tracking algorithm. The user can "link" these faces by selecting the relevant parts of the face timelines (using a standard Windows^{RTM} selection technique for

multiple items) and then clicking on a "link" screen button (not shown). The face timelines would then reflect the linkage of the whole group of face detections into one longer tracked face. Figures 21a and 21b schematically illustrate two variants of clip icons 910' and 910''. These are displayed on the display screen of Figure 20 to allow the user to select individual clips for inclusion in the time line and editing of their start and end positions (in and out points). So, each clip icon represents the whole of a respective clip stored on the system.

In Figure 21a, a clip icon 910'' is represented by a single face picture stamp 912 and a text label area 914 which may include, for example, time code information defining the position and length of that clip. In an alternative arrangement shown in Figure 21b, more than one face picture stamp 916 may be included by using a multi-part clip icon.

Another possibility for the clip icons 910 is that they provide a "face summary" so that all detected faces are shown as a set of clip icons 910, in the order in which they appear (either in the source material or in the edited output sequence). Again, faces that are the same person but which have not been associated with one another by the tracking algorithm can be linked by the user subjectively observing that they are the same face. The user could select the relevant face clip icons 910 (using a standard Windows^{RTM} selection technique for multiple items) and then click on a "link" screen button (not shown). The tracking data would then reflect the linkage of the whole group of face detections into one longer tracked face.

Figures 22a to 22c schematically illustrate a gradient pre-processing technique.

It has been noted that image windows showing little pixel variation can tend to be detected as faces by a face detection arrangement based on eigenfaces or eigenblocks. Therefore, a pre-processing step is proposed to remove areas of little pixel variation from the face detection process. In the case of a multiple scale system (see above) the pre-processing step can be carried out at each scale.

The basic process is that a "gradient test" is applied to each possible window position across the whole image. A predetermined pixel position for each window position, such as the pixel at or nearest the centre of that window position, is flagged or labelled in dependence on the results of the test applied to that window. If the test shows that a window has little pixel variation, that window position is not used in the face detection process.

A first step is illustrated in Figure 22a. This shows a window at an arbitrary window position in the image. As mentioned above, the pre-processing is repeated at each possible window position. Referring to Figure 22a, although the gradient pre-processing could be

applied to the whole window, it has been found that better results are obtained if the pre-processing is applied to a central area 1000 of the test window 1010.

Referring to Figure 22b, a gradient-based measure is derived from the window (or from the central area of the window as shown in Figure 22a), which is the average of the absolute differences between all adjacent pixels 1011 in both the horizontal and vertical directions, taken over the window. Each window centre position is labelled with this gradient-based measure to produce a gradient "map" of the image. The resulting gradient map is then compared with a threshold gradient value. Any window positions for which the gradient-based measure lies below the threshold gradient value are excluded from the face detection process in respect of that image.

Alternative gradient-based measures could be used, such as the pixel variance or the mean absolute pixel difference from a mean pixel value.

The gradient-based measure is preferably carried out in respect of pixel luminance values, but could of course be applied to other image components of a colour image.

Figure 22c schematically illustrates a gradient map derived from an example image. Here a lower gradient area 1070 (shown shaded) is excluded from face detection, and only a higher gradient area 1080 is used. The embodiments described above have related to a face detection system (involving training and detection phases) and possible uses for it in a camera-recorder and an editing system. It will be appreciated that there are many other possible uses of such techniques, for example (and not limited to) security surveillance systems, media handling in general (such as video tape recorder controllers), video conferencing systems and the like.

It will be appreciated that the embodiments of the invention described above may of course be implemented, at least in part, using software-controlled data processing apparatus. For example, one or more of the components schematically illustrated or described above may be implemented as a software-controlled general purpose data processing device or a bespoke program controlled data processing device such as an application specific integrated circuit, a field programmable gate array or the like. It will be appreciated that a computer program providing such software or program control and a storage, transmission or other providing medium by which such a computer program is stored are envisaged as aspects of the present invention.

Although illustrative embodiments of the invention have been described in detail herein with reference to the accompanying drawings, it is to be understood that the invention is not limited to those precise embodiments, and that various changes and modifications can be

effected therein by one skilled in the art without departing from the scope and spirit of the invention as defined by the appended claims.

The list of references and appendices follow. For the avoidance of doubt, it is noted that the list and the appendices form a part of the present description. These documents are hereby incorporated by reference.

References

1. H. Schneiderman and T. Kanade, "A statistical model for 3D object detection applied to faces and cars," IEEE Conference on Computer Vision and Pattern Detection, 2000.
2. H. Schneiderman and T. Kanade, "Probabilistic modelling of local appearance and spatial relationships for object detection," IEEE Conference on Computer Vision and Pattern Detection, 1998.
3. H. Schneiderman, "A statistical approach to 3D object detection applied to faces and cars," PhD thesis, Robotics Institute, Carnegie Mellon University, 2000.
4. E. Hjelmas and B.K. Low, "Face Detection: A Survey," Computer Vision and Image Understanding, no.83, pp.236-274, 2001.
5. M.-H.Yang, D.Kriegman and N.Ahuja, "Detecting Faces in Images: A Survey," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.24, no.1, pp.34-58, Jan 2002.

Appendix A: Training Face Sets

One database consists of many thousand images of subjects standing in front of an indoor background. Another training database used in experimental implementations of the above techniques consists of more than ten thousand eight-bit greyscale images of human heads with views ranging from frontal to left and right profiles. The skilled man will of course understand that various different training sets could be used, optionally being profiled to reflect facial characteristics of a local population.

Appendix B – Eigenblocks

In the eigenface approach to face detection and recognition (References 4 and 5), each *m*-by-*n* face image is reordered so that it is represented by a vector of length *mn*. Each image can then be thought of as a point in *mn*-dimensional space. A set of images maps to a collection of points in this large space.

Face images, being similar in overall configuration, are not randomly distributed in this *mn*-dimensional image space and therefore they can be described by a relatively low dimensional subspace. Using principal component analysis (PCA), the vectors that best account for the distribution of face images within the entire image space can be found. PCA involves determining the principal eigenvectors of the covariance matrix corresponding to the original face images. These vectors define the subspace of face images, often referred to as the *face space*. Each vector represents an *m*-by-*n* image and is a linear combination of the original face images. Because the vectors are the eigenvectors of the covariance matrix corresponding to the original face images, and because they are face-like in appearance, they are often referred to as *eigenfaces* [4].

When an unknown image is presented, it is projected into the face space. In this way, it is expressed in terms of a weighted sum of eigenfaces.

In the present embodiments, a closely related approach is used, to generate and apply so-called "eigenblocks" or eigenvectors relating to blocks of the face image. A grid of blocks is applied to the face image (in the training set) or the test window (during the detection phase) and an eigenvector-based process, very similar to the eigenface process, is applied at each block position. (Or in an alternative embodiment to save on data processing, the process is applied once to the group of block positions, producing one set of eigenblocks for use at any block position). The skilled man will understand that some blocks, such as a central block often representing a nose feature of the image, may be more significant in deciding whether a face is present.

Calculating Eigenblocks

The calculation of eigenblocks involves the following steps:

(1). A training set of N_T images is used. These are divided into image blocks each of size m x n. So, for each block position a set of image blocks, one from that position in each image, is obtained: $\{I_o{}^t\}_{t=1}^{N_T}$.

(2). A normalised training set of blocks $\{I^t\}_{t=1}^{N_T}$, is calculated as follows:

Each image block, $I_o{}^t$, from the original training set is normalised to have a mean of zero and an L2-norm of 1, to produce a respective normalised image block, I^t.

For each image block, $I_o{}^t$, $t = 1..N_T$:

$$I^t = \frac{I_o{}^t - \text{mean\_}I_o{}^t}{\left\| I_o{}^t - \text{mean\_}I_o{}^t \right\|}$$

$$\text{where} \quad \text{mean\_}I_o{}^t = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} I_o{}^t[i,j]$$

$$\text{and} \quad \left\| I_o{}^t - \text{mean\_}I_o{}^t \right\| = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}\left(I_o{}^t[i,j] - \text{mean\_}I_o{}^t\right)^2}$$

(i.e. the L2-norm of $\left(I_o{}^t - \text{mean\_}I_o{}^t\right)$)

(3). A training set of vectors $\{x^t\}_{t=1}^{N_T}$ is formed by lexicographic reordering of the pixel elements of each image block, I^t. i.e. Each m-by-n image block, I^t, is reordered into a vector, x^t, of length N=mn.

(4). The set of deviation vectors, $D = \{x^t\}_{t=1}^{N_T}$, is calculated. D has N rows and N_T columns.

(5). The covariance matrix, Σ, is calculated:

$$\Sigma = DD^T$$

Σ is a symmetric matrix of size N x N.

(7). The whole set of eigenvectors, P, and eigenvalues, λ_i, $i = 1,..,N$, of the covariance matrix, Σ, are given by solving:

$$\Lambda = P^T \Sigma P$$

Here, Λ is an N x N diagonal matrix with the eigenvalues, λ_i, along its diagonal (in order of magnitude) and P is an N x N matrix containing the set of N eigenvectors, each of length N. This decomposition is also known as a Karhunen-Loeve Transform (KLT).

The eigenvectors can be thought of as a set of features that together characterise the variation between the blocks of the face images. They form an orthogonal basis by which any image block can be represented, i.e. in principle any image can be represented without error by a weighted sum of the eigenvectors.

If the number of data points in the image space (the number of training images) is less than the dimension of the space ($N_T < N$), then there will only be N_T meaningful eigenvectors. The remaining eigenvectors will have associated eigenvalues of zero. Hence, because typically $N_T < N$, all eigenvalues for which $i > N_T$ will be zero.

Additionally, because the image blocks in the training set are similar in overall configuration (they are all derived from faces), only some of the remaining eigenvectors will characterise very strong differences between the image blocks. These are the eigenvectors with the largest associated eigenvalues. The other remaining eigenvectors with smaller associated eigenvalues do not characterise such large differences and therefore they are not as useful for detecting or distinguishing between faces.

Therefore, in PCA, only the M principal eigenvectors with the largest magnitude eigenvalues are considered, where $M < N_T$ i.e. a partial KLT is performed. In short, PCA extracts a lower-dimensional subspace of the KLT basis corresponding to the largest magnitude eigenvalues.

Because the principal components describe the strongest variations between the face images, in appearance they may resemble parts of face blocks and are referred to here as *eigenblocks*. However, the term *eigenvectors* could equally be used.

Face Detection using Eigenblocks

The similarity of an unknown image to a face, or its *faceness*, can be measured by determining how well the image is represented by the face space. This process is carried out on a block-by-block basis, using the same grid of blocks as that used in the training process.

The first stage of this process involves projecting the image into the face space.

Projection of an Image into Face Space

Before projecting an image into face space, much the same pre-processing steps are performed on the image as were performed on the training set:

(1). A test image block of size m x n is obtained: I_o.

(2). The original test image block, I_o is normalised to have a mean of zero and an L2-norm of 1, to produce the normalised test image block, I :

$$I = \frac{I_o - \text{mean\_}I_o}{\|I_o - \text{mean\_}I_o\|}$$

$$\text{where} \quad \text{mean\_}I_o = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} I_o[i,j]$$

$$\text{and} \quad \|I_o - \text{mean\_}I_o\| = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}(I_o[i,j] - \text{mean\_}I_o)^2}$$

(i.e. the L2-norm of $(I_o - \text{mean\_}I_o)$)

(3). The deviation vectors are calculated by lexicographic reordering of the pixel elements of the image. The image is reordered into a deviation vector, x^t, of length $N=mn$.

After these pre-processing steps, the deviation vector, x, is projected into face space using the following simple step:

(4). The projection into face space involves transforming the deviation vector, x, into its eigenblock components. This involves a simple multiplication by the M principal eigenvectors (the eigenblocks), P_i, $i = 1,..,M$. Each weight y_i is obtained as follows:

$$y_i = P_i^T x$$

where P_i is the i^{th} eigenvector.

The weights y_i, $i = 1,..,M$, describe the contribution of each eigenblock in representing the input face block.

Blocks of similar appearance will have similar sets of weights while blocks of different appearance will have different sets of weights. Therefore, the weights are used here as feature vectors for classifying face blocks during face detection.